LA-UR-98-2188

Title:

HIP 9.0 User's Guide

Author(s):

Brian J. Reardon

Submitted to:

http://lib-www.lanl.gov/la-pubs/00418677.pdf

Los Alamos
NATIONAL LABORATORY

Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by the University of California for the U.S. Department of Energy under contract W-7405-ENG-36. By acceptance of this article, the publisher recognizes that the U.S. Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. The Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; therefore, the Laboratory as an institution does not endorse the viewpoint of a publication or guarantee its technical correctness.

HIP 9.0 User's Guide

Brian J. Reardon, MST-6, Los Alamos National Laboratory, Los Alamos, NM 87545

1.0 Abstract

A computer program for modeling densification and grain growth of powders during HIPing is introduced. The phenomenological equations, upon which this program is based, are derived from the work of Ashby and the HIP6.1 program. This program differs from Ashby's in the solution of the differential equations and in the fact that actual processing schedules can be incorporated into the model. While the new program does not provide a graphical representation, it does provide grain sizes and densities as a function of time and processing schedule.

2.0 Introduction

2.1 Powder Consolidation Modeling

Powder consolidation modeling is pursued in both industrial and research environments for a variety of reasons. The two main reasons are cost and health concerns. While not all powders are necessarily expensive, the processing of powders through hot isostatic pressing (HIPing) is often costly. Additionally, the machining of parts after HIPing is cost intensive. Thus, it is hoped that through the assistance of modeling, a near net shape part can be easily produced.

The risks to both worker and environment when handling some powders is well known and can be reduced by moving to a near-net shape fabrication process guided by modeling. Such a process reduces the amount of post - HIP machining required and reduces the number of test runs needed to start up production of a new part or material.

2.2 Types of Powder Consolidation Modeling

The two main types of powder consolidation modeling used in enhancing near-net shape processing are micromechanical modeling and finite element modeling. Each technique has advantages as well as limitations.

2.2.1 Micromechanical Modeling

The micromechanical densification model first introduced by Ashby [1] and Artz [2] and later used as the basis of HIP6.1[3] creates densification maps (figure 1.) which indicate the density of a powder as a function of temperature, pressure and time. The HIP maps also provide information on the dominate densification mechanism present and the relative amount of grain growth.

While all of this information is useful, there are a number of disadvantages to using this code. First, the user can not incorporate actual HIPing schedules. Second, no quantitative information regarding grain growth is communicated. Third, since the micromechanical model is based on the local interactions of a dense random packing of monosized spheres this model can not account for powder - can interactions, density gradients, or thermal gradients, which are all observed experimentally.

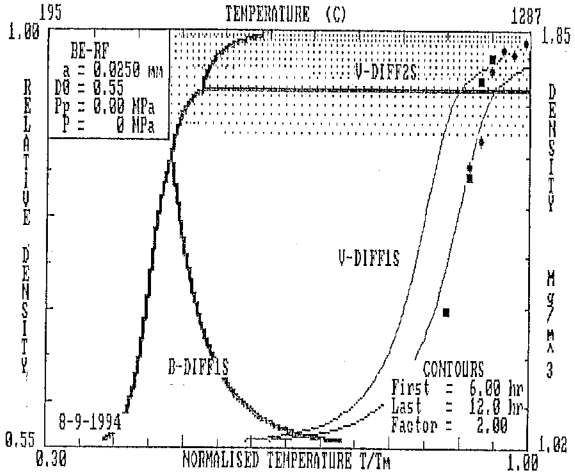

Figure 1. HIP densification map for beryllium powder calculated using Ashby's HIP6.1 from Abeln and Osborn[4].

2.2.2 Continuum - FEA

Continuum mechanics modeling is based on the mathematical constructs of finite element analysis (FEA)[5]. In FEA the structure being modeled is broken up into discrete elements which react to stress and temperature fields according to their defined material properties. FEA of powder consolidation allows for incorporation of HIP-can effects as well as thermal gradients. These two factors are usually sufficient to produce density gradients in the final part which are observed experimentally.

One former drawback to the FEA approach is that micromechanics of the consolidation could not be considered. Instead the powder was treated as a continuum. However, Li and

Häggblad's [6] recent formulation of a volumetric strain rate in terms of a densification rate offers the possibility of incorporating the micromechanical data into the continuum model. This will be done by assuming the powder continuum is always in a state of yield according to the Von-Mises yield criterion and the rate of strain is a function of the densification rate calculated micromechanically. Li and Häggblad's [6] formulation is:

$$\dot{\varepsilon}_{ij} = \frac{\sum_{i=1}^{N} K_i f_i(\rho)}{3 f I_1 \rho}\left(\frac{3}{2} c\sigma_{ij} + f I_1 \delta_{ij}\right) \qquad \text{Eq. 1.}$$

where ρ is the density, $\sum_{i=1}^{N} K_i f_i(\rho)$ is the total densification rate from all known densification mechanisms, c and f are related to the equivalent stress: $\sigma_{eq} = \left(3cJ_2 + fI_1^2\right)^{1/2}$, and J_2 and I_1 are the stress invariants definded as:

$$J_2 = \frac{1}{6}\left[\left(\sigma_{xx}-\sigma_{yy}\right)^2 + \left(\sigma_{yy}-\sigma_{zz}\right)^2 + \left(\sigma_{zz}-\sigma_{xx}\right)^2 + 6\left(\sigma_{xy}^2 + \sigma_{yz}^2 + \sigma_{zx}^2\right)\right] \qquad \text{Eq. 2.}$$

and $I_1 = \sigma_{xx} + \sigma_{yy} + \sigma_{zz}$. The beauty of such a formulation lays in the fact that with micromechanical data in hand and with temperature-pressure-density histories known throughout the part being modeled, the degree of grain growth can also be determined throughout.

2.3 Motivation for Rewriting HIP6.1

There are a number of motivating factors for rewriting Ashby's code beyond implementing it in the FEA code as described in the previous section. Additionally, the ability to incorporate actual HIP processing schedules is highly desirable. This means that densification and grain growth equations must be solved as a function of time using a fourth order Runge - Kutta method [7]. This allows for a fast evaluation of the system of differential equations with a minimal amount of error.

Other factors motivating the HIP6.1 rewrite include the desire to give the user more control of how the program handles the transition from stage I to stage II densification and the need to express the code in FORTRAN as opposed to Pascal. FORTRAN handles mathematical functions more efficiently than Pascal and allows the program to be compiled on a wider variety of machines.

Finally, there is a need to optimize the large number of micromechanical model variables such that the experimental and calculated data correlate using pre-existing optimization routines which are also written in FORTRAN[8].

3.0 Densification Mechanisms

3.1 Ashby’s Micromechanical Model

In Ashby’s micromechanical model, densification is expressed as the instantaneous change in density due to particle yielding, ρ_Y, and the total densification rate, $\dot{\rho}_T$, is a linear sum of the densification rates due to diffusive mechanisms, $\dot{\rho}_D$, power law creep, $\dot{\rho}_{PLC}$ and Nabarro-Herring creep, $\dot{\rho}_{NH}$. The model also assumes that there exists two densification stages (I and II). Stage I is characterized by open porosity whereas stage II has closed porosity.

The micromechanical formulation is based on two assumptions. The first being that the average number of contacts (Z) a particle has with other particles is a function of the relative density, Z=12ρ, and that the relative density of a dense random packing of monosized spheres is 0.64. These two assumptions, which have been experimentally verified, are maintained in the HIP 9.0 formulation.

All of the formulations in sections 3.X and 4.0 are taken directly from Ashby’s work [3].

3.2 Yielding

The instantaneous density due to yielding in Stage I is :

$$\rho_Y = \left[\frac{P\left(1-\rho_o\right)}{1.3\sigma_Y} + \rho_o^3\right]^{1/3} \qquad \text{Eq. 2.}$$

and in stage II it is:

$$\rho_Y = 1 - exp\left[\frac{-3P}{2\sigma_Y}\right] \qquad \text{Eq. 3.}$$

where P is applied pressure, σ_Y is yield stress of the powder particles, and ρ_o is the initial density.

The yield formulations are listed in file yield.f, subroutine YIELD. The YIELD subroutine requires the present temperature, pressure, and density.

3.3 Volume Diffusion

The stage I densification rate due to volume diffusion is

$$\dot{\rho}_V = 32\left(1-\rho_o\right)\frac{D_V}{r^2}\tilde{F}_1 \qquad \text{Eq. 4.}$$

and the stage II rate is:

$$\dot{\rho}_V = 3\left[\frac{1-\rho}{6\rho}\right]^{1/3} \frac{D_V}{r^2}\tilde{F}_2 \qquad \text{Eq. 5.}$$

where r is the mean particle radius, ρ is the instantaneous compact density, $\bar{F}_1$ is the stage I driving force, $\bar{F}_2$ is the stage II driving force, and D_v is the volume diffusion term.

$\bar{F}_1$ is defined as:

$$\tilde{F}_1 = \frac{(P-P_o)\Omega}{kT} + \frac{3\rho^2(2\rho-\rho_o)\gamma\Omega}{(1-\rho_o)rkT} \qquad \text{Eq. 6.}$$

where k is Boltzman's constant, T is temperature, γ is surface energy, and Ω is atomic volume;

$\bar{F}_2$ is defined as:

$$\tilde{F}_2 = \frac{(P-P_1)\Omega}{kT} + \frac{2\gamma\Omega}{rkT}\left[\frac{6\rho}{1-\rho}\right]^{1/3} \qquad \text{Eq. 7.}$$

P_1 is the internal pore pressure defined as:

$$P_1 = \left[\frac{1-\rho_c}{1-\rho}\right]\frac{\rho}{\rho_c}P_o \qquad \text{Eq. 8.}$$

with P_0 the initial pore pressure.

D_v is the volume diffusion term defined as:

$$D_V = D_{OV}\,exp\left(\frac{-Q_V}{RT}\right) \qquad \text{Eq. 9.}$$

where D_{OV} is the volume diffusion pre-exponential factor, Q_V the volume diffusion activation energy, and R is the gas constant;

The volume diffusion formulations are listed in file mechs.f, real functions VD1 and VD2.

3.4 Grain Boundary Diffusion

The stage I densification rate due to boundary diffusion is

$$\dot{\rho}_B = 43\frac{(1-\rho_o)}{(\rho-\rho_o)}\frac{\delta D_B}{r^3}\tilde{F}_1 \qquad \text{Eq. 10.}$$

and the stage II rate is:

$$\dot{\rho}_B = 4\frac{\delta D_B}{r^3}\tilde{F}_2 \qquad \text{Eq. 11.}$$

where δ is the grain boundary width, and D_B is the grain boundary diffusion term defined as:

$$D_B = D_{OB}\,exp\left(\frac{-Q_B}{RT}\right) \qquad \text{Eq. 12.}$$

where D_{OB} is the boundary diffusion pre-exponential factor, and Q_B the boundary diffusion activation energy.

The boundary diffusion formulations are listed in file mechs.f, real functions BD1 and BD2.

3.5 Power Law Creep

The stage I densification rate due to power law creep is

$$\dot{\rho}_{PLC} = 3.1\rho\left[\frac{(1-\rho_o)}{(\rho-\rho_o)}\right]^{1/2} D_C\left[\left[\frac{(1-\rho_o)}{(\rho-\rho_o)}\right]\left[\frac{P-P_o}{3\rho^2\sigma_{REF}}\right]\right]^n \qquad \text{Eq. 13.}$$

where σ_{REF} is the power law creep reference stress, n is the power law creep exponent and D_C is the creep diffusion term.

The Stage II power law creep densification rate is:

$$\dot{\rho}_{PLC} = 1.5\rho(1-\rho)D_C\left[\frac{1.5(P-P_1)}{n\sigma_{REF}\left(1-(1-\rho)^{1/n}\right)}\right]^n \qquad \text{Eq. 14}$$

The power law creep diffusion term is defined as:

$$D_C = 10^{-6}\, exp\left(\frac{-Q_C}{RT_M}\left(\frac{T_M}{T}-2\right)\right) \qquad \text{Eq. 15.}$$

where T_M is the melting temperature and Q_C the power law creep activation energy.

The powder law creep formulations are listed in file mechs.f, real functions PC1 and PC2.

3.6 Nabarro - Herring Creep

Nabarro - Herring creep is only active when the grain size is significantly smaller than the particle size. The stage I densification rate due to Nabarro-Herring creep is

$$\dot{\rho}_{NH} = \frac{14.4}{\rho}\left[\frac{(1-\rho_o)}{(\rho-\rho_o)}\right]^{1/2}\left[\frac{D_V}{G^2}+\frac{\pi\delta D_B}{G^3}\right]\tilde{F}_1 \qquad \text{Eq. 16.}$$

where G is the average grain size.

The stage II formulation is:

$$\dot{\rho}_{NH} = 32(1-\rho)\left[\frac{D_V}{G^2}+\frac{\pi\delta D_B}{G^3}\right]\tilde{F}_2 \qquad \text{Eq. 17.}$$

The Nabarro - Herring creep formulations are listed in file mechs.f, real functions RNH1 and RNH2.

4.0 Grain Growth

The grain growth rate due to pores being dragged by moving boundaries is defined as:

$$\dot{G}_{drag} = \left[1 - \frac{\bar{R}}{R_{max}}\right]\frac{2\gamma_{gb}\Omega^{4/3}}{G_o \bar{G} kT}\left[\frac{\delta D_{os}\, exp\left[\frac{-Q_s}{kT}\right] D_M\, exp\left[\frac{-Q_M}{kT}\right]}{\delta D_{os}\, exp\left[\frac{-Q_s}{kT}\right]\Omega^{1/3} + D_M\, exp\left[\frac{-Q_M}{kT}\right] 0.01(1-\rho)^{4/3} G^2}\right]$$

Eq. 18.

where D_{os} is the pre-exponential factor for surface diffusion, Q_s is the activation energy for surface diffusion, D_m pre-exponential factor for grain boundary mobility, $\bar{R}$ is the average particle radius, R_{max} is the maximum particle radius, and Q_m is the activation energy for grain boundary mobility.

If the relative density satisfies:

$$\rho > 1 - \frac{2}{9}\left[1 - \frac{\bar{R}}{R_{max}}\right]^3$$

Eq. 19

then pore detachment from the grain boundary occurs and the grain growth rate is defined as:

$$\dot{G}_{det} = \frac{2 D_M \gamma_{gb} \Omega^{2/3}}{G_o kT}\, exp\left[\frac{-Q_M}{kT}\right]\left[\left[\frac{1}{G} - \frac{\bar{R}}{G R_{max}}\right] + \frac{0.68}{\bar{R}}(1-\rho)^{2/3}\right]$$

Eq. 20

where G_o is the initial grain size.

While grain growth has no effect on the density per se, it does have an effect on the densification rate. The stage I and II volume diffusion densification rates are reduced by a factor of $\left(\frac{G_o}{2G} + \frac{1}{2}\right)$ and $\left(\frac{G_o}{G}\right)^2$ respectively. The stage I and II boundary diffusion densification rates are reduced by a factor of $\left(\frac{G_o}{G}\right)$ and $\left(\frac{G_o}{G}\right)^3$ respectively. Power law and Nabarro-Herring creep are not effected by grain growth.

5.0 HIP 9.0 Description

HIP 9.0 solves the densification and grain growth equations simultaneously as a function of time using a fourth order Runge - Kutta method with adaptive time stepping which allows for the implementation of HIPing schedules. This program also allows for the user to define when stage I ends and stage II begins.

The smoothing functions operate in the following way: If the density, ρ, $\leq$ the stage 2 cut off density, ρ_2, then the stage 1 and stage 2 weighting factors (s1, s2) are 1 and 0 respectively. If $\rho \geq$ the stage 1 cut off density, ρ_1, then s1=0 and s2=1. If $\rho_2 \leq \rho \leq \rho_1$ then $s1=(\rho_1-\rho)/(\rho_1-\rho_2)$ and s2=1-s1. All of the appropriately weighted densification rates are then added to give a total densification rate.

The grain growth formulations are listed in file graingrowth.f, real fuctions GRATE, POREDRAG and POREDETACH.

6.0 Example Problem

The example problem presented here deals with the HIPing of Tantalum powder. Once the source code is compiled using the Makefile an executable file called HIP is created. HIP reads in the two input files, data and schedule. The data file lists all of the micromechanical modeling parameters needed in the equations presented previously. The schedule file lists the HIPing schedule the powder is to be subjected to.

Once complete, HIP will create an output file called output1 that lists all of the input data and the density and grain size of the powder as a function of time.

6.1 Input Files

Input file: data

```
16600.00        read(10,14) theorden
3287.00         read(10,14) tm
180.95e-3       read(10,14) atmw
1.81e-29        read(10,14) watmv
2.3170          SURFACE ENERGY(J/M^2)
972.20          YOUNGS MODULUS(GPA)
600.50          YIELD STRESS(MPA)
0.23810         TEMPERATURE DEPENDANCE OF YEILD
5.9970          POWER LAW CREEP EXPONENT
63.610          POWER LAW CREEP REFERENCE STRESS(MPA)
398.5           POWERLAWCREEP ACTN ENERGY(KJ/MOL)
1650.0          LOW TEMP.-HIGH TEMP. CREEP TRANS. (K)
0.65170         C FOR LT CREEP
0.0053820       PRE-EXPONENT VOLUME DIFFUSION(M^2/S)'
491.91          VOLUME DIFFUSION ACTIVATION ENERGY(KJ/MOL)'
4.4930e-14      PRE-EXPONENT BOUNDRY DIFFUSION(M^2/S)'
```

```
295.14          BOUNDRY DIFF. ACT. ENERGY(KJ/MOL)'
7.6870e-10      PRE-EXPONENT SURFACE DIFFUSION(M^2/S)'
491.91          SURFACE DIFFUSION ACT. ENERGY(KJ/MOL)'
9.1670e-14      PRE-EXPONENT BOUNDRY MOBILITY(M^2/S)'
393.52          BOUNDRY MOBILITY ACT ENERGY(KJ/MOL)'
5.2420e-06      PARTICLE RADIUS(M)'
0.50    read(10,14) Rratio
1.0484e-5       read(10,14) Gdia
0.66            initial density
0.81510         Stage II cut off density
0.92430         Stage I cut off density
```

Input file: schedule

```
33200                   total time
4                       number of data points in temperature schedule
0.0000  298.00          time(s)   Temperature (K)
14400   1673.0          time(s)   Temperature (K)
28800   1673.0          time(s)   Temperature (K)
33200   298.00          time(s)   Temperature (K)
4                       number of data points in pressure schedule
0.0000  0.10000         time(s)   Pressure(MPa)
14400   207             time(s)   Pressure(MPa)
28800   207             time(s)   Pressure(MPa)
33200   0.10000         time(s)   Pressure(MPa)
```

6.2 Output File: output1

Output1 lists all of the input parameters, HIP schedule, and the density and grain size as a function of time.

```
INPUT PARAMETERS
  16600.0 THEORETICAL DENSITY (KG/M^3)
  3287.00 MELTING TEMPERATURE (K)
  .180950 ATOMIC WEIGHT(KG/MOL)
  1.810000E-29 WEIGHTED ATOMIC VOLUME(M^3/ATOM)
  2.31700 SURFACE ENERGY(J/M^2)
  972.200 YOUNGS MODULUS(GPA)
  600.500 YIELD STRESS(MPA)
  .238100 TEMPERATURE DEPENDANCE OF YEILD
  5.99700 POWER LAW CREEP EXPONENT
  63.6100 POWER LAW CREEP REFERENCE STRESS(MPA)
  398.500 POWERLAWCREEP ACTN ENERGY(KJ/MOL)
  1650.00 LOW TEMP.-HIGH TEMP. CREEP TRANS. (K)
  .651700 C FOR LT CREEP
  5.382000E-03 PRE-EXPONENT VOLUME DIFFUSION(M^2/S)
  491.910 VOLUME DIFFUSION ACTIVATION ENERGY(KJ/MOL)
  4.493000E-14 PRE-EXPONENT BOUNDRY DIFFUSION(M^2/S)
  295.140 BOUNDRY DIFF. ACT. ENERGY(KJ/MOL)
  7.687000E-10 PRE-EXPONENT SURFACE DIFFUSION(M^2/S)
  491.910 SURFACE DIFFUSION ACT. ENERGY(KJ/MOL)
  9.167000E-14 PRE-EXPONENT BOUNDRY MOBILITY(M^2/S)
  393.520 BOUNDRY MOBILITY ACT ENERGY(KJ/MOL)
  5.242000E-06 PARTICLE RADIUS(M)
  .500000 RATIO OF MEAN AND LARGEST PARTICLE SIZE
  1.048400E-05 GRAIN DIAMETER(M)
  .660000 STARTING RELATIVE DENSITY
  .815100 STAGE 2 CUT OFF RELATIVE DENSITY
  .924300 STAGE 1 CUT OFF RELATIVE DENSITY
HIP SCHEDULE
  33200.0 TOTAL PROCESS TIME
  4 TIME(S)-TEMP.(K) SCHEDULE POINTS
  .000000  298.000
  14400.0  1673.00
  28800.0  1673.00
  33200.0  298.000
  4 TIME(S)-PRESSURE(MPA) SCHEDULE POINTS
  .000000  .100000
  14400.0  207.000
  28800.0  207.000
  33200.0  .100000

OUTPUT
          TIME      DENSITY          GRAINGROWTH
  0.0000E+00  0.6600E+00  0.1048E-04
  0.1000E+01  0.6601E+00  0.1048E-04
  0.3634E+01  0.6601E+00  0.1048E-04
  0.1417E+02  0.6602E+00  0.1048E-04
  0.5631E+02  0.6605E+00  0.1048E-04
  0.1041E+03  0.6609E+00  0.1048E-04
  0.2392E+03  0.6622E+00  0.1048E-04
  0.3744E+03  0.6637E+00  0.1048E-04
  0.6508E+03  0.6673E+00  0.1048E-04
  0.9568E+03  0.6720E+00  0.1048E-04
  0.1487E+04  0.6813E+00  0.1048E-04
  0.1988E+04  0.6916E+00  0.1048E-04
  0.2961E+04  0.7132E+00  0.1048E-04
  0.3883E+04  0.7349E+00  0.1048E-04
  0.6114E+04  0.7862E+00  0.1048E-04
  0.8560E+04  0.8363E+00  0.1048E-04
  0.1132E+05  0.8876E+00  0.1048E-04
  0.1440E+05  0.9467E+00  0.1048E-04
  0.1440E+05  0.9467E+00  0.1048E-04
  0.1440E+05  0.9467E+00  0.1048E-04
  0.1440E+05  0.9468E+00  0.1048E-04
  0.1442E+05  0.9471E+00  0.1048E-04
  0.1448E+05  0.9484E+00  0.1048E-04
  0.1474E+05  0.9530E+00  0.1048E-04
  0.1577E+05  0.9645E+00  0.1048E-04
  0.1931E+05  0.9795E+00  0.1048E-04
  0.2485E+05  0.9875E+00  0.1048E-04
  0.2880E+05  0.9904E+00  0.1048E-04
  0.2880E+05  0.9904E+00  0.1048E-04
  0.2880E+05  0.9904E+00  0.1048E-04
  0.2881E+05  0.9904E+00  0.1048E-04
  0.2882E+05  0.9904E+00  0.1048E-04
  0.2889E+05  0.9904E+00  0.1048E-04
  0.2914E+05  0.9905E+00  0.1048E-04
  0.3017E+05  0.9905E+00  0.1048E-04
 0.3320E+05  0.9905E+00  0.1048E-04
```

7.0 References

[1]Ashby, M. F., “A First Report On Deformation - Mechanism Maps,” Acta Metallurgica, Vol. 20, No. 1, 1972, p. 887.

[2] Arzt, E., Ashby, M. F., Easterling, K. E., “Practical Applications Of Hot-Isostatic Pressing Diagrams : 4 Case Studies,” Metallurgical Transactions A - Physical Metallurgy And Materials Science, Vol. 14a, No. 2, 1983, pp. 211-221.

[3] Ashby, M. F., “Hip - Maps Software And Operating Manual,” Cambridge University Engineering Department, Cambridge, Uk, 1987.

[4] Abeln, S. P., Osborn, D. W., “Evaluation of Computer Models for Consolidation of Beryllium Powder,” Internal DOE Report, 1994.

[5] Bathe, K. -J., Wilson, E. L., Numerical Methods in Finite Element Analysis, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1976.

[6] Li, W. B., H.-Å.Häggblad, “Constitutive laws for hot isostatic pressing of powder compact,” Powder Metallurgy, Vol. 40, No. 4, 1997, pp. 279-281.

[7] Press, W.H., Flannery, B.P., Teukolsky, S.A., Vetterling, W.T., Numerical Recipes: The Art of Scientific Computing, Cambridge University Press, Cambridge, (1988)

[8] Reardon, B. J., “Optimization Of Micromechanical Densification Parameters For Copper Powder Using A Fuzzy Logic Based Multiobjective Genetic Algorithm,” LA-UR-98-0419, Los Alamos National Laboratory, Los Alamos, NM, 1998.

8.0 Codes

8.1 MAKEFILE

```
### MAKEFILE PREAMBLE ###

TargetName  =  HIP
Makefile    = # "{TargetName}".make
ObjDir_ppc   = :

# Power Macintosh tools and default settings

f77_ppc         = "{AbsoftTools}f77" -c
f77_ppc_opts    =  -C

# Object and resource files used for this build:

PPC_LIBS =  ∂
 "{AbsoftLibraries}MRWE.o" ∂
 "{AbsoftLibraries}libfio.o" ∂
 "{AbsoftLibraries}libfmath.o" ∂
 "{AbsoftLibraries}initFApp.o"

PPC_OBJECTS = ∂
 "{ObjDir_ppc}data.o" ∂
 "{ObjDir_ppc}derivs.o" ∂
 "{ObjDir_ppc}graingrowth.o" ∂
 "{ObjDir_ppc}HIP2.o" ∂
 "{ObjDir_ppc}initializer.o" ∂
 "{ObjDir_ppc}mechs.o" ∂
 "{ObjDir_ppc}rates.o" ∂
 "{ObjDir_ppc}rungkutta.o" ∂
 "{ObjDir_ppc}yield.o"

# Linkage rule

"{TargetName}" ƒƒ {PPC_OBJECTS} {makefile}
 Delete -i "{TargetName}"
 Duplicate -y -r "{AbsoftLibraries}"mrwe.o "{TargetName}"
 lnk -O  -aliases "{AbsoftLibraries}"absoft_aliases ∂
    -o "{TargetName}" {PPC_OBJECTS} {PPC_LIBS} ∂
    -c 'Mrwe' -t APPL ∂
    "{AbsoftLibraries}absoft_init.o" ∂
    "{PPCLibraries}StdCRuntime.o" ∂
    "{SharedLibraries}StdCLib" ∂
    "{SharedLibraries}InterfaceLib"


# Compilation rules

"{ObjDir_ppc}data.o" ƒ "data.f" {makefile}
 {f77_ppc} {f77_ppc_opts} -o "{ObjDir_ppc}data.o" data.f

"{ObjDir_ppc}derivs.o" ƒ "derivs.f" {makefile}
 {f77_ppc} {f77_ppc_opts} -o "{ObjDir_ppc}derivs.o" derivs.f

"{ObjDir_ppc}graingrowth.o" ƒ "graingrowth.f" {makefile}
 {f77_ppc} {f77_ppc_opts} -o "{ObjDir_ppc}graingrowth.o" graingrowth.f

"{ObjDir_ppc}HIP2.o" ƒ "HIP2.f" {makefile}
 {f77_ppc} {f77_ppc_opts} -o "{ObjDir_ppc}HIP2.o" HIP2.f

"{ObjDir_ppc}initializer.o" ƒ "initializer.f" {makefile}
 {f77_ppc} {f77_ppc_opts} -o "{ObjDir_ppc}initializer.o" initializer.f

"{ObjDir_ppc}mechs.o" ƒ "mechs.f" {makefile}
 {f77_ppc} {f77_ppc_opts} -o "{ObjDir_ppc}mechs.o" mechs.f

"{ObjDir_ppc}rates.o" ƒ "rates.f" {makefile}
 {f77_ppc} {f77_ppc_opts} -o "{ObjDir_ppc}rates.o" rates.f

"{ObjDir_ppc}rungkutta.o" ƒ "rungkutta.f" {makefile}
 {f77_ppc} {f77_ppc_opts} -o "{ObjDir_ppc}rungkutta.o" rungkutta.f

"{ObjDir_ppc}yield.o" ƒ "yield.f" {makefile}
 {f77_ppc} {f77_ppc_opts} -o "{ObjDir_ppc}yield.o" yield.f

clean ƒ
```

```
 Delete -i "{TargetName}" {PPC_OBJECTS}
```

8.2 File: commons

```
      COMMON /GPROP/ RGCTHEORDEN,RGCTM,RGCATMW,RGCWATMV,RGCSURNRG
      COMMON /MPROP/ RGCY,RGCSY,RGCYT,RGCPCLEXP,RGCPCLREFS,
     +          RGCPCLACTNRG,RGCLT2HT,RGCLTCC
      COMMON /DIFFPROP/ RGCPREVD,RGCACTVD,RGCPREBD,RGCACTBD
      COMMON /GGROWTH1/ RGCPRESD,RGCACTSD,RGCPREBM,RGCACTBM,RGVGSIZE
      COMMON /PARTICLE/ RGCR,RGCRRATIO,RGCGDIA
      COMMON /POWDER/ RGCDENO,PORECHECK
      COMMON /SIMDATA/ RGAPRESSSCHED(10,2),RGATEMPSCHED(10,2),
     +          RGCTOTALTIME,CUT1,CUT2
      COMMON /UNICON/ RGCBOLTZMAN,RGCGAS,RGCPI
```

8.3 File: params

8.4 File: HIP2.f

```
      PROGRAM HIP
C
      INCLUDE 'PARAMS'
      INCLUDE 'COMMONS'
      COMMON /PATH/ KMAX,KOUNT,DXSAV,XP(200),YP(10,200)

      DIMENSION YSTART(10)
C
      OPEN(UNIT=30,FILE='OUTPUT1',STATUS='OLD')
      CALL DATAINPUT(IPOINTS)
      CALL INITIALIZER(RGVGNSIZE)
          KMAX = 200
          DXSAV = 0.1
          YSTART(1) = RGCDENO
          YSTART(2) = RGVGNSIZE
          NVAR = 2
          EPS = .005
          H1 = 1.0
          HMIN = 1.0
      DO J = 1,IPOINTS-1
          X1 = RGATEMPSCHED(J,1)
          X2 = RGATEMPSCHED(J+1,1)
          CALL ODEINT(YSTART,NVAR,X1,X2,EPS,H1,HMIN,NOK,NBAD)
          DO I=1,KOUNT
              A = XP(I)
              B = YP(1,I)
              C = YP(2,I)
```

```
                        WRITE(30,30) A,B,C
                END DO
        END DO
        CLOSE(30)
 30     FORMAT(1X,5E12.4)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

        END
```

8.5 File: data.f

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCC SUBROUTINE DATAINPUT
CCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
    SUBROUTINE DATAINPUT(POINTS)
    INCLUDE 'PARAMS'
    INCLUDE 'COMMONS'
C
        INTEGER POINTS

        WRITE(30,*) 'INPUT PARAMETERS'
        OPEN(UNIT=10,FILE='DATA',STATUS='OLD')
CC GENERAL PROPERTIES
        READ(10,*) RGCTHEORDEN
        WRITE(30,*) RGCTHEORDEN, 'THEORETICAL DENSITY (KG/M^3)'
         READ(10,*) RGCTM
        WRITE(30,*) RGCTM, 'MELTING TEMPERATURE (K)'
        READ(10,*) RGCATMW
        WRITE(30,*) RGCATMW,'ATOMIC WEIGHT(KG/MOL)'
        READ(10,*) RGCWATMV
        WRITE(30,*)  RGCWATMV, 'WEIGHTED ATOMIC VOLUME(M^3/ATOM)'
        READ(10,*) RGCSURNRG
        WRITE(30,*) RGCSURNRG, 'SURFACE ENERGY(J/M^2)'
CC  MECHANICAL PROPERTIES
        READ(10,*) RGCY
        WRITE(30,*)RGCY, 'YOUNGS MODULUS(GPA)'
        READ(10,*) RGCSY
        WRITE(30,*)RGCSY, 'YIELD STRESS(MPA)'
         READ(10,*) RGCYT
        WRITE(30,*)RGCYT, 'TEMPERATURE DEPENDANCE OF YEILD'
        READ(10,*) RGCPCLEXP
        WRITE(30,*)RGCPCLEXP, 'POWER LAW CREEP EXPONENT'
        READ(10,*) RGCPCLREFS
        WRITE(30,*)RGCPCLREFS, 'POWER LAW CREEP REFERENCE STRESS(MPA)'
        READ(10,*) RGCPCLACTNRG
        WRITE(30,*)RGCPCLACTNRG, 'POWERLAWCREEP ACTN ENERGY(KJ/MOL)'
        READ(10,*) RGCLT2HT
        WRITE(30,*)RGCLT2HT,'LOW TEMP.-HIGH TEMP. CREEP TRANS. (K)'
        READ(10,*) RGCLTCC
        WRITE(30,*)RGCLTCC, 'C FOR LT CREEP'
CC  DIFFUSION PROPERTIES
        READ(10,*) RGCPREVD
        WRITE(30,*)RGCPREVD, 'PRE-EXPONENT VOLUME DIFFUSION(M^2/S)'
        READ(10,*) RGCACTVD
        WRITE(30,*)RGCACTVD,'VOLUME DIFFUSION ACTIVATION ENERGY(KJ/MOL)'
        READ(10,*) RGCPREBD
        WRITE(30,*)RGCPREBD, 'PRE-EXPONENT BOUNDRY DIFFUSION(M^2/S)'
        READ(10,*) RGCACTBD
        WRITE(30,*)RGCACTBD,'BOUNDRY DIFF. ACT. ENERGY(KJ/MOL)'
CC  GRAIN GROWTH
        READ(10,*) RGCPRESD
        WRITE(30,*)RGCPRESD, 'PRE-EXPONENT SURFACE DIFFUSION(M^2/S)'
        READ(10,*) RGCACTSD
        WRITE(30,*)RGCACTSD, 'SURFACE DIFFUSION ACT. ENERGY(KJ/MOL)'
        READ(10,*) RGCPREBM
        WRITE(30,*)RGCPREBM, 'PRE-EXPONENT BOUNDRY MOBILITY(M^2/S)'
        READ(10,*) RGCACTBM
        WRITE(30,*)RGCACTBM, 'BOUNDRY MOBILITY ACT ENERGY(KJ/MOL)'
CC PARTICLE CHARACTORISTICS
        READ(10,*) RGCR
        WRITE(30,*)RGCR, 'PARTICLE RADIUS(M)'
        READ(10,*) RGCRRATIO
        WRITE(30,*) RGCRRATIO,'RATIO OF MEAN AND LARGEST PARTICLE SIZE'
        READ(10,*) RGCGDIA
        WRITE(30,*)RGCGDIA, 'GRAIN DIAMETER(M)'
        READ(10,*) RGCDENO
        WRITE(30,*)RGCDENO, 'STARTING RELATIVE DENSITY'
        READ(10,*) CUT1
        WRITE(30,*)CUT1, 'STAGE 2 CUT OFF RELATIVE DENSITY'
        READ(10,*) CUT2
        WRITE(30,*)CUT2, 'STAGE 1 CUT OFF RELATIVE DENSITY'
CCCCCCCCCCCCCCCCCCCCCCCCCCC
        CLOSE(10)

        OPEN(UNIT=10,FILE='SCHEDULE',STATUS='OLD')

        WRITE(30,*) 'HIP SCHEDULE'
        READ(10,*) RGCTOTALTIME
        WRITE(30,*)RGCTOTALTIME, 'TOTAL PROCESS TIME'

        READ(10,*) POINTS
        WRITE(30,*)POINTS, 'TIME(S)-TEMP.(K) SCHEDULE POINTS'
        DO 10 J = 1,POINTS
                READ(10,*) X2,Y2
                WRITE(30,*)X2,Y2
                RGATEMPSCHED(J,1)=X2
                RGATEMPSCHED(J,2)=Y2
 10     CONTINUE

        READ(10,*) POINTS
        WRITE(30,*)POINTS, 'TIME(S)-PRESSURE(MPA) SCHEDULE POINTS'
         DO 30 J = 1,POINTS
        READ(10,*) X2,Y2
                WRITE(30,*)X2,Y2
        RGAPRESSSCHED(J,1)=X2
                RGAPRESSSCHED(J,2)=Y2
 30     CONTINUE
        CLOSE(10)

        WRITE(30,*)''
        WRITE(30,*)'OUTPUT'
        WRITE(30,*)'TIME            DENSITY'
        RETURN
        END
```

8.6 File: initializer.f

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCC SUBROUTINE INITIALIZER
CCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
        SUBROUTINE INITIALIZER(RGVGNSIZE)
        INCLUDE 'PARAMS'
        INCLUDE 'COMMONS'

        RGCBOLTZMAN = 1.381E-23
        RGCGAS = 8.314
        RGCPI = ACOS(-1.0)
        RGVGNSIZE = RGCGDIA
        PORECHECK = 1-(2./9.)*(1-RGCRRATIO)**3
        RETURN
        END
```

8.6 File: rungkutta.f

```
SUBROUTINE ODEINT(YSTART,NVAR,X1,X2,EPS,H1,HMIN,NOK,NBAD)
C       Ranger-Kutta driver with adaptive stepsize control.  Integrate the NVAR starting
C       values YSTART from X1 to X2 with accuracy EPS, storing intermediate results in
C       the common block /PATH/.  H1 should be set as a guessed first stepsize, HMIN as
C       the minimum allowed stepsize (can be zero).  On output NOK and NBAD are the
C       number of good and bad (but retried and fixed) steps taken, and YSTART is
C       replaced by values at the end of the integration interval.  DERIVS is the
C       user-supplied subroutine for caclulating the right-hand side derivative, while RKQC
C       is the name of the stepper routine to be used.  PATH contains its own information
C       about how often an intermediate value is to be stored.
        PARAMETER (MAXSTP=10000,NMAX=10,TWO=2.0,ZERO=0.0,TINY=1.E-30)
         COMMON /PATH/ KMAX,KOUNT,DXSAV,XP(200),YP(10,200)
C       User storage for intermediate results.  Preset DXSAV and KMAX.
        DIMENSION YSTART(NVAR),YSCAL(NMAX),Y(NMAX),DYDX(NMAX)
        X=X1
        H=SIGN(H1,X2-X1)
         NOK=0
         NBAD=0
        KOUNT=0
        DO 11 I=1,NVAR
                Y(I)=YSTART(I)
 11     CONTINUE
        IF (KMAX.GT.0) XSAV=X-DXSAV*TWO
         DO 16 NSTP=1,MAXSTP
C       CALL DERIVS(X,Y,DYDX)
        CALL DERIVS(0.,X,Y,DYDX)
        DO 12 I=1,NVAR
                YSCAL(I)=ABS(Y(I))+ABS(H*DYDX(I))+TINY
 12     CONTINUE
        IF(KMAX.GT.0) THEN
                IF(ABS(X-XSAV).GT.ABS(DXSAV)) THEN
                        IF(KOUNT.LT.KMAX-1) THEN
                                KOUNT=KOUNT+1
                                XP(KOUNT)=X
```

```
                                  DO 13 I=1,NVAR
                                          YP(I,KOUNT)=Y(I)
 13                               CONTINUE
                                  XSAV=X
                          ENDIF
                  ENDIF
          ENDIF
          IF((X+H-X2)*(X+H-X1).GE.ZERO) H=X2-X
          CALL RKQC(Y,DYDX,NVAR,X,H,EPS,YSCAL,HDID,HNEXT)
          IF(HDID.EQ.H)THEN
                  NOK=NOK+1
          ELSE
                  NBAD=NBAD+1
          ENDIF
          IF((X-X2)*(X2-X1).GE.ZERO)THEN
                  DO 14 I=1,NVAR
                          YSTART(I)=Y(I)
 14               CONTINUE
                  IF(KMAX.NE.0)THEN
                          KOUNT=KOUNT+1
                          XP(KOUNT)=X
                          DO 15 I=1,NVAR
                                  YP(I,KOUNT)=Y(I)
 15                       CONTINUE
                  ENDIF
                  RETURN
          ENDIF
C         IF(ABS(HNEXT).LT.HMIN) PAUSE 'Stepsize smaller then minimum.'
          H=HNEXT
 16   CONTINUE
C     PAUSE 'Too many steps.'
      RETURN
      END

      SUBROUTINE RKQC(Y,DYDX,N,X,HTRY,EPS,YSCAL,HDID,HNEXT)
C         Fifth-order Runge-Kutta step with monitoring of local truncation error to ensure
C         accuracy and adjust stepsize.  Input are the dependent variable vector Y of length N
C         and its derivative DYDX at the starting value of the independent variable X.  Also
C         input are the stepsize to be attempted HTRY, the required accuracy EPS, and the
C         vector YSCAL against which the error is scaled.  On output, Y and X are replaced
C         by their new values , HDID is the stepsize which was actually accomplished, and
C         HNEXT is the estimated next stepsize.  DERIVS is the user-supplied subroutine
C         that computes the right-hand side derivatives.
      PARAMETER (NMAX=10,PGROW=-0.20,PSHRNK=-0.25,FCOR=1./15.,
     *ONE=1.,SAFETY=0.9,ERRCON=6.E-4)
C         The value ERRCON equals (4/SAFETY)**(1/PGROW)
      DIMENSION Y(N),DYDX(N),YSCAL(N),YTEMP(NMAX),YSAV(NMAX),
     *DYSAV(NMAX)
      XSAV=X
      DO 11 I=1,N
          YSAV(I)=Y(I)
          DYSAV(I)=DYDX(I)
 11   CONTINUE
      H=HTRY
 1    HH=0.5*H
      CALL RK4(YSAV,DYSAV,N,XSAV,HH,YTEMP)
      X=XSAV+HH
C     CALL DERIVS(X,YTEMP,DYDX)
```

```
      CALL DERIVS(XSAV,X,YTEMP,DYDX)
      CALL RK4(YTEMP,DYDX,N,X,HH,Y)
      X=XSAV+H
C     IF(X.EQ.XSAV) PAUSE 'STEP SIZE NOT SIGNIFICANT IN RKQC.'
      CALL RK4(YSAV,DYSAV,N,XSAV,H,YTEMP)
      ERRMAX=0
      DO 12  I=1,N
            YTEMP(I)=Y(I)-YTEMP(I)
            ERRMAX=MAX(ERRMAX,ABS(YTEMP(I)/YSCAL(I)))
 12   CONTINUE
      ERRMAX=ERRMAX/EPS
      IF(ERRMAX.GT.ONE) THEN
            H=SAFETY*H*(ERRMAX**PSHRNK)
            GOTO 1
      ELSE
            HDID=H
            IF(ERRMAX.GT.ERRCON) THEN
                      HNEXT=SAFETY *H*(ERRMAX**PGROW)
            ELSE
                      HNEXT=4.*H
            ENDIF
      ENDIF
      DO 13  I=1,N
            Y(I)=Y(I)+YTEMP(I)*FCOR
13    CONTINUE
      RETURN
      END


      SUBROUTINE RK4(Y,DYDX,N,X,H,YOUT)
C           Given values for N variables Y and their derivatives DYDX known at X, use the
C           fourth-order Range-Kutta method to advance the solution over an interval H and
C           return the incremented variables as YOUT, which need not be a distinct array from
C           Y.  The user supplies the subroutine DERIVS(X,Y,DYDX) which returns
C           derivatives DYDX at X.
      PARAMETER (NMAX=10)
      DIMENSION Y(N),DYDX(N),YOUT(N),YT(NMAX),DYT(NMAX),DYM(NMAX)
      HH=H*0.5
      H6=H/6.
      XH=X+HH
      DO 11 I=1,N
            YT(I)=Y(I)+HH*DYDX(I)
 11   CONTINUE
C     CALL DERIVS(XH,YT,DYT)
      CALL DERIVS(X,XH,YT,DYT)
      DO 12 I=1,N
            YT(I)=Y(I)+HH*DYT(I)
 12   CONTINUE
C     CALL DERIVS(XH,YT,DYM)
      CALL DERIVS(X,XH,YT,DYM)
      DO 13 I=1,N
            YT(I)=Y(I)+H*DYM(I)
            DYM(I)=DYT(I)+DYM(I)
 13   CONTINUE
C     CALL DERIVS(X+H,YT,DYT)
      CALL DERIVS(X,X+H,YT,DYT)
      DO 14 I=1,N
            YOUT(I)=Y(I)+H6*(DYDX(I)+DYT(I)+2.*DYM(I))
 14   CONTINUE
ccccccccccccccccccccccc
    if (YOUT(1).GE.1.0) YOUT(1)=1.0

  RETURN

  END
```

8.7 File: derivs.f

```
      SUBROUTINE DERIVS(OLDT,NEWT,Y,DYDT)

      REAL NEWT,OLDT

      INCLUDE 'COMMONS'

      PARAMETER (NMAX=10,ZERO=0.0,TINY=1.E-30)
      INCLUDE 'PARAMS'

      DIMENSION Y(NMAX),DYDT(NMAX)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      IF (Y(1).GT.1.0) Y(1) = 1.0
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      DO J = 1,NMAX-1
            IF ((NEWT.GE.RGATEMPSCHED(J,1)).AND.
  *           (NEWT.LT.RGATEMPSCHED(J+1,1))) THEN
              X1 = RGATEMPSCHED(J,1)
              X2 = RGATEMPSCHED(J+1,1)
              Y1 = RGATEMPSCHED(J,2)
              Y2 = RGATEMPSCHED(J+1,2)
              SLOPE = (Y2-Y1)/(X2-X1)
              B = Y1-SLOPE*X1
              TEMPERATURE = SLOPE*NEWT + B
     ENDIF
            IF ((NEWT.GE.RGAPRESSSCHED(J,1)).AND.
  *           (NEWT.LT.RGAPRESSSCHED(J+1,1))) THEN
              X1 = RGAPRESSSCHED(J,1)
              X2 = RGAPRESSSCHED(J+1,1)
              Y1 = RGAPRESSSCHED(J,2)
              Y2 = RGAPRESSSCHED(J+1,2)
              SLOPE = (Y2-Y1)/(X2-X1)
              B = Y1-SLOPE*X1
              PRESSURE = SLOPE*NEWT + B
     ENDIF
      END DO
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      IF (Y(1).GE.1.0) THEN
            DYDT(1) = ZERO
      ELSE
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C TIME INDEPENDENT YIELDING EFFECTS
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
            TDENSITY=Y(1)
            CALL YIELD(TEMPERATURE,PRESSURE,TDENSITY)
            DDEN = TDENSITY-Y(1)

            DT = (NEWT-OLDT)
            IF (DT.LE.TINY) THEN
                  YIELDRATE = ZERO
            ELSE
                  YIELDRATE = DDEN/DT
            ENDIF
            IF(YIELDRATE.LE.0.0) YIELDRATE=0.0
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C TIME DEPENDENT DENSIFICATION MECHANISMS
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
            DYDT(1) = 0.0
            CALL RATES(TEMPERATURE,PRESSURE,TDENSITY,Y(2),
  *         RATE,DYDT(2))
            DYDT(1)=RATE+YIELDRATE
      ENDIF
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C GRAIN GROWTH RATE EFFECTS
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      DYDT(2) = 0.0
      DYDT(2)=GRATE(TEMPERATURE,TDENSITY,Y(2))
      RETURN
      END
```

8.7 File: yield.f

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCC SUBROUTINE YIELD
CCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
   SUBROUTINE YIELD(T,P,D)
   INCLUDE 'PARAMS'
   INCLUDE 'COMMONS'
C
      REAL P,PN,T,TN,D,T_FACTOR,RM,SAT,C1,C2,WK_HDNG,
  &         PN_T,DSTAGE1,DSTAGE2,DEN
C
      PN = P/RGCSY
      TN = T/RGCTM

            T_FACTOR = (1-RGCYT*(TN*RGCTM-293)/RGCTM)
            IF (T_FACTOR.LT.0.02) T_FACTOR = 0.02
C
            IF (D.LT.RGCDENO) D = RGCDENO
            RM = 0.5
            SAT = 2.4
            C2=1+(EXP(LOG(1+(SAT-1)*(1.01-TN))/RM)-1)
  &                     *((D-RGCDENO)/(1-RGCDENO))**2.
            WK_HDNG = EXP(LOG(C2)*RM)
C

            PN_T = PN/(T_FACTOR*WK_HDNG)
            C1 = (1-RGCDENO)*PN_T/1.3 + RGCDENO*RGCDENO*RGCDENO
            DSTAGE1 = EXP(LOG(C1)/3.)

            IF (PN_T.LE.15) DSTAGE2 = 1-EXP(-1.5*PN_T)
            IF (PN_T.GT.15) DSTAGE2 = 1.0

            IF (DSTAGE1.GT.0.95) DSTAGE1 = 0.95
```

```
           DEN = DSTAGE1
           IF (DSTAGE2.GT.DSTAGE1) DEN = DSTAGE2

           IF (DEN.LT.D) DEN = D
           D=DEN
           IF (D.GE.1.0) D=1.0
40   CONTINUE
   RETURN
   END
```

8.8 File: rates.f

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCC SUBROUTINE RATES
CCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
   SUBROUTINE RATES(T,P,D,GNOW,RGVTOTALRATE,GRATE)
   INCLUDE 'PARAMS'
   INCLUDE 'COMMONS'

      REAL P,T,DEN1,FP,FS,S1,TEST

      DIMENSION RATE(15)
   RGVTOTALRATE = 0.0
      DEN1 = D

      IF (DEN1.LT.1.0) THEN
           GRATIO=RGCGDIA/GNOW
           IF (D.GT.PORECHECK) THEN
                 GBD = GRATIO
                 GVD = (GRATIO+1.)/2.
           ELSE
                 GBD = (GRATIO)**3
                 GVD = (GRATIO)**2
           ENDIF
           DO 10 I = 1,15
                 RATE(I) = 0.0
10         CONTINUE

           IF ((DEN1.GE.CUT1)) THEN
                 FP=F2P(P,T)
                 FS=F2S(P,T,DEN1)

                 S1 = 1.0
                 SLOPE = (1.-0.)/(1.-CUT1)
                 B = 0.-SLOPE*CUT1
                 S1= SLOPE*DEN1 + B
                 IF (S1.LT.0.) S1 = 0.

                 RATE(1)=PC2(P,T,DEN1,S1)
                 RATE(2)=BD2(T,FP,S1)*GBD
                 RATE(3)=BD2(T,FS,S1)*GBD
                 RATE(4)=VD2(T,DEN1,FP,S1)*GVD
                 RATE(5)=VD2(T,DEN1,FS,S1)*GVD
                 RATE(6)=RNH2(T,DEN1,FP,S1,GNOW)
                 RATE(7)=RNH2(T,DEN1,FS,S1,GNOW)
           ENDIF
```

```
               IF (DEN1.LE.CUT2) THEN
                     FP=F1P(P,T)
                     FS=F1S(T,DEN1)

                     S1=1.0
                     SLOPE = (0.-1.)/(CUT2-RGCDENO)
                     B = 1.-SLOPE*RGCDENO
                     S1 = SLOPE*DEN1 + B
                     IF (S1.LT.0.) S1 = 0.

               RATE(8) =PC1(P,T,DEN1,S1)
               RATE(9) =BD1(T,DEN1,FP,S1)*GBD
                     RATE(10)=BD1(T,DEN1,FS,S1)*GBD
                     RATE(11)=VD1(T,FP,S1)*GVD
                     RATE(12)=VD1(T,FS,S1)*GVD
                     RATE(13)=RNH1(T,DEN1,FP,S1,GNOW)
                     RATE(14)=RNH1(T,DEN1,FS,S1,GNOW)
               ENDIF
               RGVTOTALRATE = 0.
               DO 35 K = 1,14
                     TEST = RATE(K)
                     IF (TEST.LT.0.) TEST = 0.
                     RGVTOTALRATE = RGVTOTALRATE+TEST
 35            CONTINUE
         ENDIF

      RETURN
      END
```

8.9 File: graingrowth.f

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCC SUBROUTINE GRAIN
CCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      REAL FUNCTION GRATE(T,D,GNOW)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      INCLUDE 'PARAMS'
      INCLUDE 'COMMONS'
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
         GRATE = 0.0
         IF (D.GT.PORECHECK) THEN
               GRATE=POREDETACH(T,D,GNOW)
         ELSE
               GRATE=POREDRAG(T,D,GNOW)
         ENDIF
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCC SUBROUTINE POREDETACH
CCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      REAL FUNCTION POREDETACH(T,D,GNOW)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      INCLUDE 'PARAMS'
      INCLUDE 'COMMONS'
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
            C1=2.0
            C2=0.68
            C3=1-RGCRRATIO
            C4=RGCPREBM/(RGCGDIA**2 * RGCWATMV**(1./3.))
                  C8 =1000.*RGCACTBM/(RGCGAS*T)
            C5=EXP(-C8)
            C6=RGCSURNRG*RGCWATMV/(RGCBOLTZMAN*T)
            C7=(C3*RGCGDIA/GNOW)-(C2*RGCGDIA*(1-D)**(2./3))/RGCR
                  IF (C7.LT.0.0) C7=0.
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
            POREDETACH=C1*C4*C5*C6*C7
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCC SUBROUTINE POREDRAGGING
CCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      REAL FUNCTION POREDRAG(T,D,GNOW)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      INCLUDE 'PARAMS'
      INCLUDE 'COMMONS'
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
            GRATE=0.0
            RLIM = 60.
            ICOUNT = 0
                  C8=1000.*RGCACTSD/(RGCGAS*T)
 20               IF (C8.GT.RLIM) C8=RLIM
            C1=RGCPRESD*EXP(-C8)
                  C9=1000.*RGCACTBM/(RGCGAS*T)
                  IF (C9.GT.RLIM) C9=RLIM
            C2=((RGCWATMV)**(1./3.))*RGCPREBM*EXP(-C9)
            C3=1-RGCRRATIO
            C4=0.01
            C5=2*C3*RGCSURNRG*RGCWATMV/(RGCGDIA*GNOW*RGCBOLTZMAN*T)
            C6=C1*(RGCWATMV**(2./3.))
            C7=C4*C2*(GNOW**2)*((1.-D)**(4./3.))
            IF ((C6+C7).EQ.0.0) THEN
                  RLIM=RLIM/2.
                  IF (ICOUNT.EQ.10) THEN
                        PRINT*,'WARNING IN POREDRAG'
                        STOP
                  ENDIF
                  ICOUNT = ICOUNT+1
                  GOTO 20
            ENDIF
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
            POREDRAG=C5*C1*C2/(C6+C7)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      RETURN
      END
```

8.10 File: mechs.f

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCC FUNCTION F2P
CCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      REAL FUNCTION F2P(P,T)
      INCLUDE 'PARAMS'
      INCLUDE 'COMMONS'
      F2P = 0.0
      F2P = 1E6*P*RGCWATMV/(RGCBOLTZMAN*T)
      IF (F2P.LE.0.) F2P=0.
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCC FUNCTION F2S
CCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      REAL FUNCTION F2S(P,T,DEN1)
      INCLUDE 'PARAMS'
      INCLUDE 'COMMONS'
      F2S=0.0

      C4 = 1-DEN1
            IF (C4.LE.0.0) C4=1E-30
      C1 = (6*DEN1/C4)**(1./3.)
      C2 = RGCSURNRG/RGCR
      C3 = RGCWATMV/(RGCBOLTZMAN*T)
      F2S=2*C1*C2*C3
      IF (F2S.LE.0.) F2S=0.
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCC FUNCTION F1P
CCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      REAL FUNCTION F1P(P,T)
      INCLUDE 'PARAMS'
      INCLUDE 'COMMONS'
      F1P=0.0
      C1 = RGCWATMV/(RGCBOLTZMAN*T)
      F1P = 1E6*P*C1
      IF (F1P.LT.0.0) F1P = 0.
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCC FUNCTION F1S
CCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      REAL FUNCTION F1S(T,DEN1)
      INCLUDE 'PARAMS'
      INCLUDE 'COMMONS'
       F1S=0.0
      C1 = (2*DEN1-RGCDENO)/(1-RGCDENO)
      C2 = RGCSURNRG/RGCR
      C3 = RGCWATMV/(RGCBOLTZMAN*T)
      F1S = 3*DEN1*DEN1*C1*C2*C3
      IF (F1S.LE.0.0) F1S=0.0
      RETURN
      END
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCC FUNCTION PC1
CCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      REAL FUNCTION PC1(P,T,DEN1,S)
      INCLUDE 'PARAMS'
      INCLUDE 'COMMONS'
      PC1=0.0
      C1 = 1000.*RGCPCLACTNRG/(RGCGAS*RGCTM)
      C2 = (RGCTM/T) - 2.
      C3 = C1*C2
      IF (C3.GT.65) C3=65
      DCH = (1E-6)*EXP(-C3)

      C1 = RGCTM/RGCLT2HT
      IF (C1.LT.1.0) C1 = 1.
      IF (C1.GT.20.) C1 = 20.
      FN_OF_C1 = RGCLTCC * ((RGCTM/T) - C1) + (C1-2)*(1+2/RGCPCLEXP)
      C3 = C1*FN_OF_C1
      IF (C3.GT.65) C3=65
      DCL = (1E-6)*EXP(-C3)

      C1 = SQRT((DEN1-RGCDENO)/(1-RGCDENO))
      C3 = P/(3*DEN1*DEN1*RGCPCLREFS)
      PH = S*3.1*DEN1*C1*DCH*EXP( RGCPCLEXP   *LOG( C3/(C1*C1) ))
      PL = S*3.1*DEN1*C1*DCL*EXP((RGCPCLEXP+2)*LOG( C3/(C1*C1) ))
      PC1 = PH + PL
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCC FUNCTION PC2
CCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      REAL FUNCTION PC2(P,T,DEN1,S)
      INCLUDE 'PARAMS'
      INCLUDE 'COMMONS'
      PC2 = 0.0

      C1 = 1000.*RGCPCLACTNRG/(RGCGAS*RGCTM)
      C2 = RGCTM/T - 2.
      C3 = C1*C2
      IF (C3.GT.65) C3=65
      DCH = (1E-6)*EXP(-C3)

      C1 = RGCTM/RGCLT2HT
      IF (C1.LT.1.0) C1 = 1.
      IF (C1.GT.20.) C1 = 20.
      FN_OF_C1 = RGCLTCC * ((RGCTM/T) - C1) + (C1-2)*(1+2/RGCPCLEXP)
      C3 = C1*FN_OF_C1
      IF (C3.GT.65) C3=65
      DCL = (1E-6)*EXP(-C3)

      C1H = 1.5*DEN1*(1-DEN1)*DCH
      C1L = 1.5*DEN1*(1-DEN1)*DCL
      C2H = 1.5/ RGCPCLEXP
      C2L = 1.5/(RGCPCLEXP+2)
      IF (DEN1.GE.1.0) THEN
            C3H = 1.0
            C3L = 1.0
      ELSE
            C3H = 1-EXP(LOG(1-DEN1)/ RGCPCLEXP   )
      C3L = 1-EXP(LOG(1-DEN1)/(RGCPCLEXP+2))
      ENDIF
      C4H = P/(RGCPCLREFS*C3H)
      C4L = P/(RGCPCLREFS*C3L)

      PH= S*C1H*EXP( RGCPCLEXP   *LOG(C2H*C4H))
      PL= S*C1L*EXP((RGCPCLEXP+2)*LOG(C2L*C4L))

      PC2 = PH + PL
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCC FUNCTION BD2
CCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      REAL FUNCTION BD2(T,F,S)
      INCLUDE 'PARAMS'
      INCLUDE 'COMMONS'
      BD2=0.0
      C1 = 1000.*RGCACTBD/(RGCGAS*T)
      IF (C1.GT.65.) C1=65.
      DELDB = RGCPREBD*EXP(-C1)

      BD2=S*4.*F*DELDB/(RGCR*RGCR*RGCR)
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCC FUNCTION BD1
CCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      REAL FUNCTION BD1(T,DEN1,F,S)
      INCLUDE 'PARAMS'
      INCLUDE 'COMMONS'
      BD1=0.0
      C1 = 1000.*RGCACTBD/(RGCGAS*T)
      IF (C1.GT.65) C1=65.
      DELDB = RGCPREBD*EXP(-C1)

      C1 = (1.-RGCDENO)/(DEN1-RGCDENO)
      BD1 = S*43.*C1*DELDB*F/(RGCR*RGCR*RGCR)
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCC FUNCTION VD2
CCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      REAL FUNCTION VD2(T,DEN1,F,S)
      INCLUDE 'PARAMS'
      INCLUDE 'COMMONS'
      VD2=0.0
      C1 = 1000.*RGCACTVD/(RGCGAS*T)
      IF (C1.GT.65.) C1=65.
      DELVD = RGCPREVD*EXP(-C1)

      C1 = ((1.-DEN1)/(6.*DEN1))**(1./3.)
      VD2=S*3.*C1*F*DELVD/(RGCR*RGCR)
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCC FUNCTION VD1
CCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      REAL FUNCTION VD1(T,F,S)
      INCLUDE 'PARAMS'
      INCLUDE 'COMMONS'
      VD1=0.0
      C1 = 1000.*RGCACTVD/(RGCGAS*T)
      IF (C1.GT.65.) C1=65.
      DELVD = RGCPREVD*EXP(-C1)
      VD1=S*32.*(1.-RGCDENO)*F*DELVD/(RGCR*RGCR)
      RETURN
       END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCC FUNCTION RNH2
CCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      REAL FUNCTION RNH2(T,DEN1,F,S,GNOW)
      INCLUDE 'PARAMS'
      INCLUDE 'COMMONS'
      RNH2=0.0

      X = RGCR*(((DEN1-RGCDENO)/(1-RGCDENO))**(0.5))/SQRT(3.)
      IF (GNOW.LE.2.*X) THEN

      C1 = 1000.*RGCACTVD/(RGCGAS*T)
      IF (C1.GT.65.) C1=65.
      DELVD = RGCPREVD*EXP(-C1)

      C1 = 1000.*RGCACTBD/(RGCGAS*T)
      IF (C1.GT.65.) C1=65.
      DELDB = RGCPREBD*EXP(-C1)

      C1=(DELVD/(GNOW*GNOW))+(RGCPI*DELDB/(GNOW*GNOW*GNOW))
      RNH2=S*32.*(1.-DEN1)*F*C1
      ENDIF
      RETURN
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCC FUNCTION RNH1
CCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      REAL FUNCTION RNH1(T,DEN1,F,S,GNOW)
      INCLUDE 'PARAMS'
      INCLUDE 'COMMONS'
      RNH1=0.0
      X = RGCR*(((DEN1-RGCDENO)/(1-RGCDENO))**(0.5))/SQRT(3.)
      IF (GNOW.LE.2.*X) THEN
      C1 = 1000.*RGCACTVD/(RGCGAS*T)
      IF (C1.GT.65.) C1=65.
      DELVD = RGCPREVD*EXP(-C1)

      C1 = 1000.*RGCACTBD/(RGCGAS*T)
      IF (C1.GT.65.) C1=65.
```

```
DELDB = RGCPREBD*EXP(-C1)

C1 = 14.4/DEN1
C2 = SQRT((1.-RGCDENO)/(DEN1-RGCDENO))
C3=(DELVD/(GNOW*GNOW))+(RGCPI*DELDB/(GNOW*GNOW*GNOW))

RNH1=S*C1*C2*C3*F
ENDIF
RETURN
END
```